Technology
Arts Sciences
TH Köln

Course
PI1 - Practical Informatics 1

Version: 2 | Last Change: 10.09.2019 15:49 | Draft: 0 | Status: vom verantwortlichen Dozent freigegeben

General information

Long name	Practical Informatics 1
Approving CModule	PI1_BaET
Responsible	Prof. Dr. Dieter Rosenthal Professor Fakultät IME
Level	Bachelor
Semester in the year	winter semester
Duration	Semester
Hours in self-study	60
ECTS	5
Professors	Prof. Dr. Dieter Rosenthal Professor Fakultät IME Ursula Derichs Lehrkraft für besondere Aufgaben
Requirements	none
Language	German
Separate final exam	Yes

Final exam

Details

Written exam:
Students shall prove that they can 1.) explain and apply fundamental terms, 2.) apply programming and more abstract concepts to solve application problems and 3.) assess the correctness of proposed solutions. Typical types of assignments are 1.) multiple choice questions, fill-in-the-blank texts, assessment of statements, 2.) solving given problems of limited size by programs and Nassi-Shneiderman diagrams and 3.) finding errors in given programs.

Minimum standard

At least 50% of the total number of points.

Exam Type

Written exam:
Students shall prove that they can 1.) explain and apply fundamental terms, 2.) apply programming and more abstract concepts to solve application problems and 3.) assess the correctness of proposed solutions. Typical types of assignments are 1.) multiple choice questions, fill-in-the-blank texts, assessment of statements, 2.) solving given problems of limited size by programs and Nassi-Shneiderman diagrams and 3.) finding errors in given programs.

Lecture / Exercises

Learning goals

Knowledge

algorithms
characteristics of algorithms
description of algorithms

---

digital computers
bits/bytes
structure of the hard- and software architecture

---

basic concepts of programming
high-level programming languages vs. machine languages
compilation vs. interpretation
procedural vs. object-oriented languages: C vs. C++

---

basic concepts of variables

---

scalar data types in C
numbers
value ranges
representation of constants
operations
characters
coding standards: ASCII, Unicode
operations
character strings
boolean values
representation of constants
operations

---

control structures in Java (und C)
abstract representation
Nassi-Shneiderman diagrams
flow charts

blocks
conditional statements
if
if-else
switch-case
loops
pre-test loops
for
while
post-test loops: do-while

---

arrays in C
indexing and loops
multi-dimensional arrays

---

functions
structure
paarameter passing (Call by value, Call by reference)

---

storage organisation:
pointer
dynamic memory allocation

---

struct in C
structure
implementation (static/dynamic)

Skills

writing algorithms to solve given problems (in natural language and in graphical form - Nassi-Shneiderman diagrams, flow charts)

---

programming with elementary operations in a higher programming language

---

programming with control structures

---

programming with functions

---

programming with structured data types like arrays and structs

Expenditure classroom teaching

Type	Attendance (h/Wk.)
Lecture	2
Exercises (whole course)	1
Exercises (shared course)	1

Tutorial (voluntary)	0

Separate exam

none

Practical training

Learning goals

Knowledge

programming elementary operations on scalar variables

programming with control structures (including the design of Nassi-Shneiderman diagrams or flow charts)

programming with structured data, esp. arrays

Skills

working with a software development environment

finding and correcting errors in programs

designing algorithms and implementing them in a higher language

application of the aspects listed above to real-world scenarios in small teams

Expenditure classroom teaching

Type	Attendance (h/Wk.)
Practical training	1
Tutorial (voluntary)	0

Separate exam

Exam Type

working on practical scenarion (e.g. in a lab)

Details

Students work in small teams. Each team completes multiple "rounds" with assigned appointments in the lab. In each round, programming assigments of an algorithmic and object-oriented nature are solved - firstly by a more abstract representation (e.g. description of an algorithm by a Nassi-Shneiderman diagram), secondly by an runnable implementation (e.g.
C-program).
For the preparation of a laboratory appointment a "preparation sheet" has to be solved. The acquired knowledge will be tested at the beginning of the appointment (short written entrance test, interview with the supervisor). In case of failure, a follow-up appointment must be taken; in case of multiple failures, the student will be excluded from the lab. In case of success, a "laboratory work sheet" with further tasks will be worked on under supervision (and, if necessary, with assistance).

Minimum standard

Successful participation in all laboratory appointments, i.e. in particular independent solution (or with some assistance if necessary) of the programming assignments.